Software Requirements Specification for

Design of Automated Resume Ranking and Role Matching System

Version 1.0 approved

Prepared by

Akhil Agrawal (A2305222080)

Ashutosh Garg (A2305222091)

Aryan Nair (A2305222105)

Organization:

Amity School of Engineering and Technology, Noida

Date: September 2025

Table of Content

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to define the functional and non-functional requirements of the *Automated Resume Ranking and Role Matching System*.
The system aims to automate the process of evaluating and ranking resumes based on their relevance to specific job descriptions using Artificial Intelligence (AI) and Natural Language Processing (NLP) techniques.
This document serves as a reference for developers, testers, project supervisors, and other stakeholders involved in the design, development, and maintenance of the system.

1.2 Document Conventions

- The document follows the **IEEE 830** standard for SRS documentation.
- Headings and subheadings are numbered hierarchically (e.g., 1, 1.1, 1.2).
- The following typographic conventions are used:
  - *Italics* for file names, modules, and UI elements.
  - **Bold** for key system components or emphasis.
  - Monospace `text` for code or commands.
- Acronyms used:
  - **AI** – Artificial Intelligence
  - **NLP** – Natural Language Processing
  - **ATS** – Applicant Tracking System
  - **UI** – User Interface

1.3 Intended Audience and Reading Suggestions

This document is intended for the following readers:

- **Project Supervisors and Reviewers** – to assess the technical completeness of the system.
- **Developers and Designers** – to use the detailed requirements for coding and system design.
- **Testers** – to create validation and verification test cases based on functional specifications.
- **End Users (Recruiters/HR Teams)** – to understand the expected functionality and workflow.

Readers should begin with **Section 1** (Introduction) for context, move to **Section 4** (Requirements) for system behavior, and refer to later sections for design and testing references.

1.4 Stakeholder Management

Effective stakeholder management ensures smooth coordination and communication throughout the project lifecycle.
The key stakeholders and their responsibilities are summarized below:

Table 1 Stakeholder Management

Stakeholder	Role/Designation	Organization / Domain
Project Members / Student Team	Developers / Researchers	Department of Computer Science and Engineering, Amity University
Academic Supervisor / Project Guide	Faculty Mentor	Department of Computer Science and Engineering, Amity University
Evaluation Committee / Examiners	Academic Reviewers	University Evaluation Board
Institution / Department	Administrative and Academic Authority	Amity School of Engineering and Technology (ASET)
End Users (Recruiters / HR Professionals)	Primary Users	Recruitment and Hiring Sector

Stakeholder communication will occur weekly through progress reports and milestone reviews, ensuring transparency and alignment with academic objectives.

1.5 Product Scope

The *Automated Resume Ranking and Role Matching System* is a web-based platform that leverages AI and NLP to automate the screening of candidate resumes and match them with job descriptions.
Key capabilities include:

- Resume upload (PDF/DOCX formats).
- AI-based extraction of candidate skills, education, and experience.
- Semantic comparison between job requirements and candidate profiles.
- Ranked display of candidate-job matches via an interactive, user-friendly interface.

The product aims to:

- **Reduce manual workload** for recruiters.
- **Improve accuracy and fairness** by minimizing human bias.
- **Enhance scalability** in handling large applicant volumes.
- **Accelerate recruitment** by automating initial shortlisting.

1.6 References

1. Project Report – *Design of Automated Resume Ranking and Role Matching System*, Amity University Uttar Pradesh, 2025.
2. Lin et al. (2016). *Machine Learned Resume-Job Matching Solution.* arXiv:1607.07657
3. Zhu et al. (2018). *Person-Job Fit Neural Network (PJFNN).* arXiv:1810.04040
4. Qin et al. (2020). *Enhanced Neural Network Approach to Person-Job Fit in Talent Recruitment (TAPJFNN).* ACM TOIS.
5. Devlin et al. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.*
6. Wang et al. (2022). *Person-Job Fit Estimation with Co-attention Neural Networks (PJFCANN).* arXiv.

2. Overall Description

2.1 Product Perspective

The *Automated Resume Ranking and Role Matching System* is designed as a **web-based AI-driven recruitment tool** that automates the evaluation of resumes and matches them against predefined job descriptions.
It functions as a **standalone intelligent application**, but it can also be integrated with existing **Applicant Tracking Systems (ATS)** or HR management software in the future.

The system adopts a **client–server architecture**, comprising:

- **Frontend (React + TypeScript):** Provides an intuitive interface for recruiters and applicants to upload resumes and view ranked results.
- **Backend (Gemini AI API):** Handles natural language processing, skill extraction, and semantic similarity analysis between resumes and job descriptions.
- **Optional Database Layer:** May be added for persistent storage of resume data, ranking results, and analytics in extended versions.

This modular architecture ensures scalability, maintainability, and easy future integration with other enterprise systems.

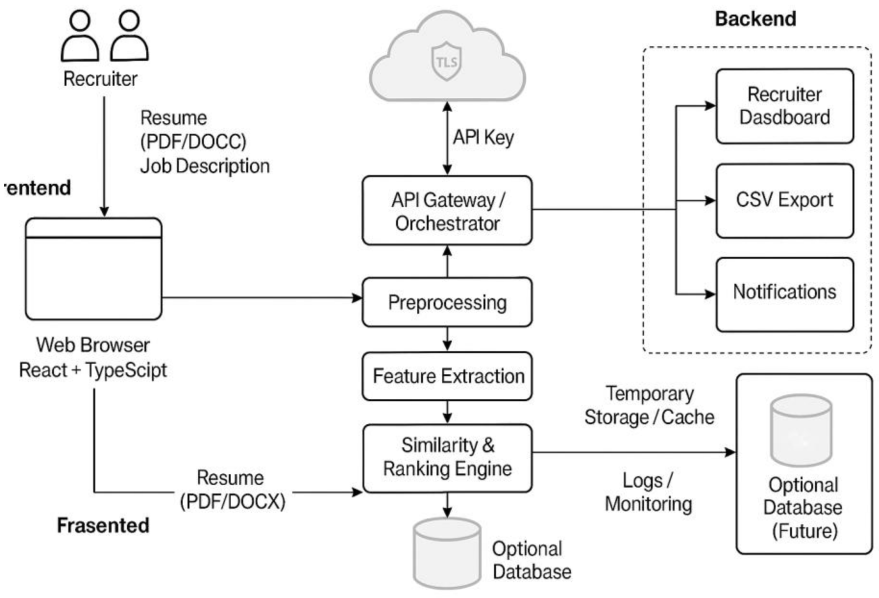

Figure 1 System Architecture and Data Flow for Automated Resume Ranking and Role Matching System

2.2 Product Functions

The main functions of the system include:

1. **Resume Upload:**
   Users can upload resumes in supported formats (PDF or DOCX) through a drag-and-drop interface.
2. **Job Role Selection:**
   Recruiters can select predefined job roles or upload custom job descriptions.
3. **Resume Parsing and Extraction:**
   The system extracts information such as skills, education, and work experience using AI and NLP models.
4. **Resume–Job Matching:**
   The backend analyzes semantic similarity between candidate attributes and job requirements to calculate a match score.
5. **Candidate Ranking:**
   Generates a ranked list of candidates based on match percentage and displays it visually using progress bars and scorecards.

6. **Result Visualization:**
   Interactive dashboards show ranking details, matched/missing skills, and candidate summaries.
7. **Bias and Transparency Controls:**
   Includes features like *Blind Mode* (hiding names/institutions) and *How We Scored* explanations to ensure fairness and accountability.

2.3 User Classes and Characteristics

The primary users of the system are:

Table 2 User Classes and Characteristics

User Class	Description	Characteristics
Recruiters / HR Managers	Upload resumes, view rankings, and shortlist candidates.	Require easy navigation, accuracy, and visualization tools.
Job Applicants (Optional)	Upload their own resumes for self-evaluation.	Expect feedback on strengths, weaknesses, and job fit.
System Administrators / Developers	Manage system updates, APIs, roles, and configurations.	Need technical control, logs, and configuration options.
Faculty/Project Evaluators (Academic Use)	Evaluate functionality, documentation, and reports.	Require understanding of workflow and outputs.

2.4 Operating Environment

The system operates in the following environment:

- **Frontend Platform:** Web browser (Google Chrome, Microsoft Edge, Mozilla Firefox).
- **Operating System:** Compatible with Windows, Linux, and macOS.
- **Backend / Processing:** Hosted online via Gemini AI API (cloud-based).
- **Development Tools:**
  - React.js, TypeScript, Tailwind CSS (Frontend)
  - Node.js (for local testing and execution)
  - Gemini API (AI processing and NLP analysis)
- **Hardware Requirements:**
  - Minimum 4 GB RAM, Dual-Core Processor
  - Internet connection for API communication

2.5 Design and Implementation Constraints

1. **Dependency on External APIs:** The system depends on Google's Gemini API, which may have usage limits, costs, and network latency.
2. **Supported File Formats:** Only PDF and DOCX resumes are supported; scanned image-based resumes cannot be processed.
3. **Single-Resume Processing:** Current version supports one resume upload at a time (batch processing planned for future).
4. **Bias in AI Models:** AI outputs may reflect biases present in training datasets.
5. **Data Privacy:** Resumes contain personal information, requiring secure handling and non-persistence.
6. **Network Dependency:** The system requires stable internet connectivity for AI-based analysis.

2.6 User Documentation

The following user documentation will be provided with the system:

- **User Manual / Quick Start Guide:** Explains installation, login, and workflow steps (resume upload → analysis → result viewing).
- **Online Help Section:** Integrated tooltips and UI instructions within the web app.
- **Developer Documentation:** Includes setup commands, API key configuration, and codebase structure.
- **Video Demonstration:** Short demo video (3–5 minutes) showing all major features.

All documentation will be version-controlled and regularly updated to align with new releases.

2.7 Assumptions and Dependencies

Assumptions

- Users will have stable internet connectivity to access the web-based platform.
- The application will run on modern web browsers supporting current front-end technologies (React, TypeScript, Tailwind).
- All resumes submitted will be in machine-readable formats such as PDF or DOCX.
- Job descriptions will be well-defined, structured, and detailed enough for AI-based analysis.
- The Gemini API or equivalent AI model will remain operational and available during system use.

- Authorized personnel will securely manage API keys and authentication credentials.

Dependencies

- **Third-Party APIs:** The system relies on the Gemini API for NLP-based resume analysis and ranking.
- **Browser Compatibility:** Proper system performance depends on browsers that support modern JavaScript frameworks.
- **Input Data Quality:** Accuracy of ranking results depends on the quality and completeness of resumes and job descriptions provided.
- **Institutional Infrastructure:** Hosting servers, network configurations, and data security measures must be managed by the institution or organization deploying the system.
- **External Libraries:** React, TypeScript, and Tailwind CSS dependencies must be regularly updated to maintain compatibility.
- **Future Integrations:** Planned features such as multilingual support, cloud storage, and ATS integration depend on third-party APIs and backend expansion.

3. External Interface Requirements

3.1 User Interfaces

The *Automated Resume Ranking and Role Matching System* provides an intuitive, responsive, and user-friendly graphical interface developed using **React** and **Tailwind CSS**. The design follows accessibility and usability guidelines to ensure smooth operation for both recruiters and applicants.

Interface Features:

1. **Home/Upload Page:**
   - Allows users to upload resumes in PDF/DOCX format via drag-and-drop or file selection.
   - Displays instructions, supported file types, and upload progress indicators.
2. **Job Role Selection Panel:**
   - Users can choose predefined job roles (e.g., Backend Developer, Data Scientist) from a dropdown list.
   - Displays job role details, skill requirements, and eligibility criteria.
3. **Analysis Page:**
   - Shows a loading spinner during resume processing.
   - Displays progress bars and status messages indicating real-time analysis.

4. **Results Display Dashboard:**
    - Presents candidate-job match percentage visually using progress bars and color coding (Green: Strong Match, Yellow: Moderate, Red: Weak).
    - Includes "How We Scored" section explaining the scoring breakdown (Skills, Experience, Education).
    - Offers **Blind Mode** to hide personal identifiers and reduce bias.
5. **Error & Notification Panel:**
    - Displays error messages for invalid file uploads, missing API keys, or connectivity issues.
    - Uses pop-up notifications and tooltips for user assistance.
6. **Accessibility:**
    - Keyboard navigation support, ARIA labels, and color contrast compliant with **WCAG 2.1 standards**.
    - Responsive design adaptable to desktop, tablet, and mobile screens.

3.2 Hardware Interfaces

The system does not require any specialized hardware components. It operates through standard computing devices with internet access.

Minimum Hardware Requirements:

Table 3 Hardware Interface

Component	**Specification**
Processor	Dual-Core 2.0 GHz or higher
RAM	Minimum 4 GB (8 GB recommended)
Storage	500 MB free space for browser cache and local logs
Display	Minimum 1280 × 720 resolution
Network	Stable broadband internet connection (minimum 2 Mbps)

The backend processing (Gemini API) is cloud-hosted, thus eliminating the need for on-premise server hardware.

3.3 Software Interfaces

The system interacts with various software components for functionality, development, and integration purposes.

Primary Software Interfaces:

Table 4 Primary Software Interfaces

Software Component	Description
Frontend Framework (React + TypeScript)	Builds the UI components for uploading, analyzing, and displaying results.
Gemini API (AI Model)	Performs NLP-based resume parsing, semantic similarity analysis, and ranking.
Node.js Runtime	Executes frontend build scripts and development server.
Tailwind CSS	Handles responsive and accessible UI styling.
Browser Environment	Supports Chrome, Edge, or Firefox for running the web interface.
Optional Database Layer	Future enhancement for storing candidate and ranking data.
Version Control (GitHub)	Used for source code management and project versioning.

All software components communicate using RESTful API calls and JSON data exchange.

3.4 Communication Interfaces

The system uses standard internet communication protocols to ensure reliable and secure interaction between the frontend, backend, and external services.

Communication Specifications:

1. **Protocol:**
   - HTTPS for all API requests and responses to ensure encrypted data transfer.
   - RESTful architecture for communication between client and server components.
2. **Data Exchange Format:**
   - JSON (JavaScript Object Notation) for structured, lightweight, and easily parseable data transmission.
3. **External API Communication:**
   - Requests are sent to the **Gemini AI API** for resume parsing and semantic analysis.
   - Includes authentication through secure API keys stored in environment variables (`.env.local`).

4. **Error Handling and Retry Mechanism:**
   - Automatic retry on timeouts and 5xx server errors using exponential backoff.
   - Graceful error messages displayed to the user for failed API requests.
5. **Network Requirements:**
   - Continuous internet connectivity is required for real-time resume processing.
   - Offline access supports only viewing pre-fetched results or help documentation.

4. System Features

This section outlines the major system features and functionalities integrated into the *Automated Resume Ranking and Role Matching System*. Each feature includes its description, purpose, inputs, processing steps, outputs, and constraints.

4.1 Feature 1: Web-Based Accessibility

Description:
The system is designed as a browser-based application accessible from any device with internet connectivity. Recruiters and applicants can upload resumes, view ranked outputs, and download analytical reports without installing additional software.

Functional Details:

- Accessible via standard browsers such as Chrome, Edge, and Firefox.
- Responsive interface compatible with laptops, tablets, and smartphones.
- Ensures consistent performance and visual experience across platforms.

Inputs: Resume file (PDF/DOCX).
Outputs: Ranked analysis and visualization results displayed on the user interface.
Constraints: Requires stable internet connectivity; not functional in offline mode.

4.2 Feature 2: Resume Classification

Description:
The system automatically classifies uploaded resumes into relevant job categories (e.g., Data Science, Web Development, Machine Learning) using NLP-based analysis. This assists recruiters in identifying candidates suited for specific domains.

Functional Details:

- Uses AI-driven semantic similarity models to determine role category.

- Enables role-specific ranking and domain-based filtering.
- Improves candidate shortlisting speed and recruiter decision accuracy.

Inputs: Extracted text from resumes.
Outputs: Job role label or category classification.
Constraints: Classification accuracy depends on resume completeness and model precision.

4.3 Feature 3: Real-Time Notifications

Description:
The system provides immediate on-screen notifications and status updates to users for key events such as successful uploads, ranking completion, or system errors.

Functional Details:

- Pop-up alerts for invalid file formats or incomplete uploads.
- Notification banners for ranking completion or API errors.
- Interactive prompts for connectivity issues or missing fields.

Inputs: User interactions and system responses.
Outputs: On-screen messages and status indicators.
Constraints: Depends on browser compatibility and active internet connection.

4.4 Feature 4: Cloud Connectivity

Description:
The application operates entirely on a cloud-enabled infrastructure, ensuring seamless communication between frontend and AI backend through secure APIs.

Functional Details:

- Utilizes HTTPS for encrypted data transmission.
- Connects to the Gemini API for real-time NLP-based analysis.
- Supports multiple users simultaneously through scalable architecture.

Inputs: Resume data and job description parameters.
Outputs: Processed similarity scores and ranked candidate lists.
Constraints: System performance depends on API uptime and internet stability.

4.5 Feature 5: Data Preprocessing

Description:
Before AI-based evaluation, the system preprocesses extracted text from resumes to standardize and clean data for reliable analysis.

Functional Details:

- Removes redundant symbols, line breaks, and inconsistent formatting.
- Normalizes abbreviations (e.g., “JS” → “JavaScript”).
- Structures extracted text into analyzable tokens for NLP processing.

Inputs: Raw text extracted from uploaded resumes.
Outputs: Clean, structured data ready for feature extraction.
Constraints: Performance varies with document formatting and text clarity.

4.6 Feature 6: Feature Extraction

Description:
The system employs NLP and AI algorithms to identify and extract crucial resume features, forming the foundation of ranking and matching.

Functional Details:

- Extracts entities such as skills, education, work experience, and certifications.
- Identifies years of experience and domain expertise using context-aware parsing.
- Generates structured candidate profiles aligned with job requirements.

Inputs: Preprocessed resume data.
Outputs: Structured feature set (skills, experience, education, etc.).
Constraints: Limited performance for image-based or poorly formatted resumes.

4.7 Feature 7: Data Logging and Evaluation

Description:
The system logs session data and processing metrics to ensure transparency, performance tracking, and debugging support.

Functional Details:

- Records filename, processing time, and candidate match scores.

- Logs API request/response times for analysis.
- Allows export of log summaries for academic evaluation or report generation.

Inputs: System process metrics and runtime events.
Outputs: Log files and CSV summaries.
Constraints: Logs are stored temporarily; no user-identifiable data is retained.

4.8 Feature 8: Recruiter Dashboard

Description:
The Recruiter Dashboard serves as the main interface for viewing ranked candidates, analyzing reports, and making data-driven hiring decisions.

Functional Details:

- Displays sortable tables of candidate names, skills, and ranking scores.
- Offers search, filter, and export-to-CSV functionalities.
- Provides graphical insights such as skill-matching charts and score distributions.

Inputs: Ranked candidate data from AI analysis.
Outputs: Interactive visualization and downloadable reports.
Constraints: Accessible only to authenticated recruiter sessions.

4.9 Feature 9: Security and Reliability

Description:
The system ensures data confidentiality, ethical AI use, and consistent operational reliability throughout the recruitment process.

Functional Details:

- Employs HTTPS and API key authentication for data protection.
- Implements automatic retry and fail-safe mechanisms for API calls.
- Includes bias mitigation checks and fairness validation in ranking.

Inputs: User credentials, resume data, and API requests.
Outputs: Securely processed and unbiased ranking results.
Constraints: Dependent on third-party API stability and security policies.

5. Other Non-Functional Requirements

5.1 Performance Requirements

The system must ensure efficient performance under normal and expected operating conditions.
Key performance criteria include:

1. **Response Time:**
   - Resume analysis and result generation must complete within **5–10 seconds** for a single resume.
2. **Concurrent Usage:**
   - The system should support multiple users uploading resumes simultaneously without performance degradation.
3. **Scalability:**
   - Designed to handle an increasing number of resumes as user base grows, with optional integration of batch processing.
4. **Resource Utilization:**
   - The application should operate smoothly on devices with 4 GB RAM or higher without excessive CPU or memory consumption.
5. **Availability:**
   - The system should maintain **99% uptime** during hosted operation, except during scheduled maintenance.

5.2 Safety Requirements

Safety requirements ensure the protection of user data, system stability, and ethical AI behavior.

1. **Data Privacy:**
   - The system does not permanently store uploaded resumes or personal user data; temporary data is cleared after session completion.
2. **Error Handling:**
   - Proper validation and fail-safe messages prevent system crashes and unexpected shutdowns.
3. **AI Reliability:**
   - The Gemini AI API is used responsibly, ensuring the model's outputs remain free of harmful or discriminatory biases.
4. **Backup and Recovery:**

- Configuration files and codebase are version-controlled through GitHub for data recovery in case of software failure.

5. **Safe Termination:**
    - If network failure occurs during analysis, the system safely terminates the process without corrupting data or output.

5.3 Security Requirements

Security mechanisms are crucial to maintain data integrity and confidentiality throughout the system lifecycle.

1. **Authentication:**
    - Secure API key authentication used for all Gemini API communications.
2. **Data Encryption:**
    - All transmitted data (resume files, job descriptions, and results) is encrypted using **HTTPS (SSL/TLS)** protocols.
3. **Access Control:**
    - Only authorized users (recruiters or administrators) can access ranking dashboards or results.
4. **Input Validation:**
    - Strict validation for uploaded files to prevent malicious file uploads (only `.pdf` and `.docx` formats accepted).
5. **Log Protection:**
    - No sensitive information such as email IDs or phone numbers is included in logs or exports.
6. **API Usage Monitoring:**
    - Request throttling and error handling mechanisms ensure protection from denial-of-service (DoS) or misuse.

5.4 Software Quality Attributes

The quality of the system is defined through the following key attributes:

Table 5 Software Quality Attributes

Attribute	Description
Usability	Simple and intuitive user interface designed with accessibility (WCAG 2.1 compliance).
Reliability	Consistent performance across multiple use cases and devices with minimal downtime.

Attribute	Description
Maintainability	Modular architecture (React components) allows easy updates and debugging.
Portability	Fully web-based and compatible with multiple operating systems and browsers.
Scalability	Can be extended to handle batch resume uploads and large organizational datasets.
Efficiency	Optimized code ensures quick response times and low memory consumption.
Testability	Each module (Uploader, Analysis, Results Display) can be individually tested for accuracy.

5.5 Business Rules

1. Only **authorized recruiters or HR users** can access detailed ranking and candidate reports.
2. The system **does not store** or reuse personal resume data after session termination.
3. AI-generated scores serve as **recommendations**, not final hiring decisions.
4. **Ethical compliance** must be maintained — the model should not use personal identifiers (gender, name, etc.) to influence results.
5. Any system updates or new role configurations must be validated through internal testing before deployment.
6. Versioning and changelog documentation are mandatory for all project updates.

6. System Features and Constraints

1. **Web-Based Accessibility:** Accessible via browsers on any device; requires stable internet connection.
2. **AI-Powered Resume Analysis:** Uses Gemini API for NLP-based matching; depends on external API uptime.
3. **Automatic Resume Classification:** Categorizes resumes into job domains; accuracy varies with data quality.
4. **Real-Time Ranking:** Generates instant candidate scores; minor delays possible under heavy load.
5. **Recruiter Dashboard:** Displays ranked results and reports; restricted to authenticated users.
6. **Data Preprocessing & Extraction:** Cleans and structures resume text; limited for image-based files.

7. **Secure Cloud Connectivity:** Ensures encrypted communication; non-functional offline.
8. **Scalability & Responsiveness:** Adapts to all devices; older browsers may face compatibility issues.
9. **Temporary Data Logging:** Records processing metrics briefly; no long-term data storage.
10. **Ethical & Reliable Operation:** Implements bias control; minor model bias may still exist.

7. Testing Plan (Mapping with FR + TP)

7.1 Test Objectives

- Verify that all **Functional Requirements (FRs)** are correctly and reliably implemented.
- Validate **Non-Functional Requirements (NFRs)** like performance, usability, and security.
- Ensure the complete workflow **(Upload → Analyze → Rank → Export)** functions as intended.
- Provide measurable and demonstrable test outcomes for evaluation.

7.2 Test Scope

In Scope:

- Web UI (React-based interface)
- Resume parsing and analysis via Gemini API
- Scoring, ranking, and visualization modules
- Bias controls (Blind Mode, "How We Scored")
- CSV export, logging, and error handling

Out of Scope:

- Batch uploads
- ATS/persistent database integration
- Multilingual or image-based parsing

7.3 Test Approach

Table 6 Test Approach

Test Type	Focus Area	Tool / Success Criteria
Unit Tests	Keyword mapping, score calculations	Jest / Vitest – All utility functions return expected results
Integration Tests	UI ↔ Gemini API ↔ Output	Playwright / Cypress – Smooth data flow, proper error handling
System / E2E Tests	Full user journey	Manual & automated – Upload to export verified
Usability & Accessibility	UI responsiveness, ARIA, keyboard nav	Lighthouse ≥90
Performance Tests	Resume processed within 10s	Stable output under normal load
Security Tests	File-type & API validation	HTTPS enforced, no PII leakage

8. Resource Allocation

8.1 Software Resources Allocation

To manage time, cost, and computing efficiency, resources are allocated as follows:

Table 7 Software Resources Allocation

Resource Category	Tool / Platform	Purpose / Allocation
Frontend Development	React + TypeScript	User-interface creation and data visualization.
Backend / AI Integration	Gemini API (Google AI Model)	Resume parsing, NLP processing, and semantic ranking.
Version Control	Git + GitHub	Repository management, issue tracking, and version history.
Testing & Debugging Tools	Node.js, VS Code, Lighthouse	Code execution, UI testing, and accessibility audits.
Documentation	MS Word / Google Docs	Report writing, SRS, SDS, and final submission documents.

Resource Category	Tool / Platform	Purpose / Allocation
Project Management	Trello / Excel	Task scheduling, progress tracking, and deadline management.
Cloud Hosting (Optional)	Vercel / Netlify	Deployment and live demonstration hosting.

Resource distribution ensures each developer manages a dedicated component — **Frontend (UI)**, **Backend (API Integration)**, and **Documentation & Testing** — enabling parallel and efficient development.

8.1.1 Work Breakdown Structure (WBS)

The Work Breakdown Structure (WBS) of the *Automated Resume Ranking and Role Matching System* divides the project into five main phases — Requirement Analysis, System Design, Implementation, Testing, and Documentation.
In the Requirement Analysis phase, project objectives, data sources, and ranking parameters are defined.
System Design focuses on creating the architecture, data flow, and integration plan for the Gemini API.
Implementation involves developing the frontend (React), backend (Flask), and connecting the AI module for resume analysis.
Testing ensures accuracy, performance, and proper communication between modules through unit and integration testing.
Documentation records all procedures, diagrams, and user instructions for clarity and maintenance.
Each phase is logically connected to ensure systematic progress and controlled development.
The WBS ensures efficient time management, task delegation, and quality assurance.
It provides a clear roadmap for project execution from planning to deployment.
Overall, it enhances project monitoring, accountability, and structured development.

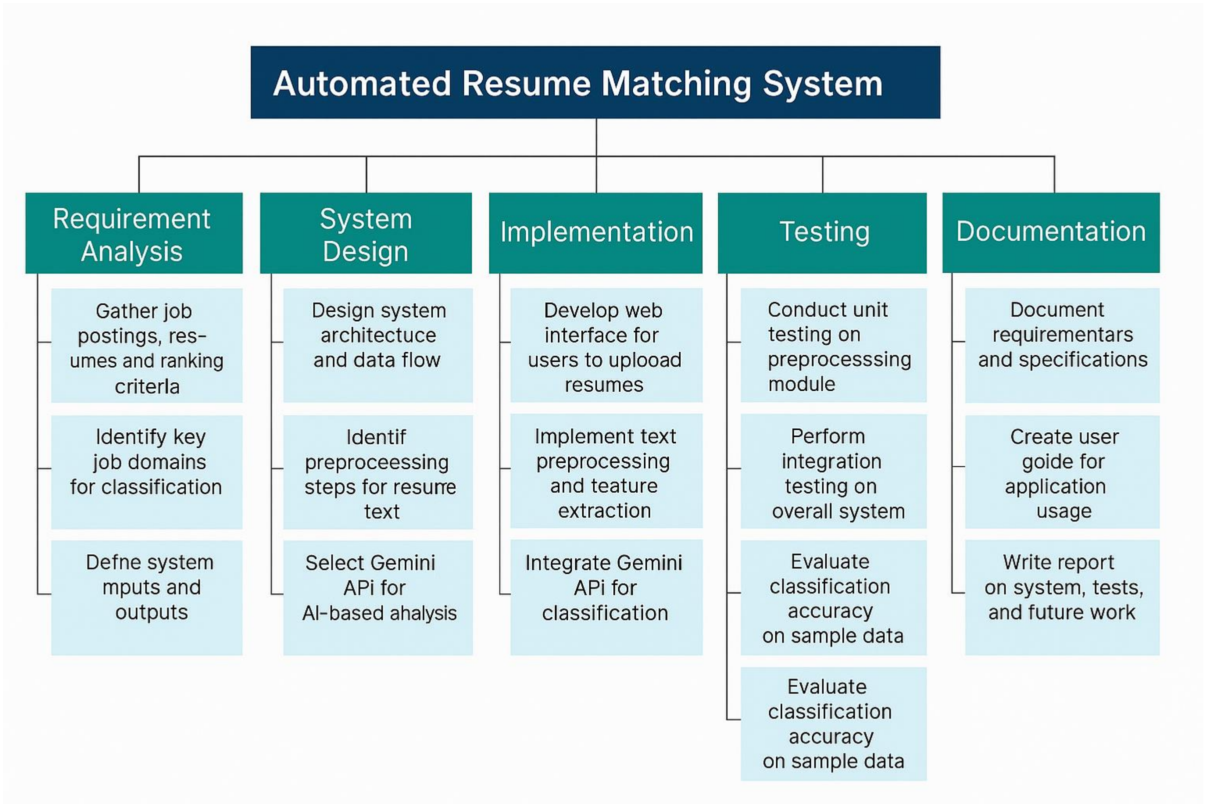
Automated Resume Matching System
Requirement Analysis
Gather job postings, res-umes and ranking criterla
Identify key job domains for classification
Defne system mputs and outputs
System Design
Design system architecture and data flow
Identif preproceessing steps for resume text
Select Gemini APi for AI-based ahalysis
Implementation
Develop web interface for users to uploaad resumes
Implement text preprocessing and teature extraction
Integrate Gemini APi for classification
Testing
Conduct unit testing on preprocesssing module
Perform integration testing on overall system
Evaluate classification accuracy on sample data
Evaluate classification accuracy on sample data
Documentation
Document requirementars and specifications
Create user goide for application usage
Write report on system, tests, and future work

Figure 2: Work Breakdown Structure (WBS) for Design of Automated Ranking and Role Matching System

8.12 Critical Path Method (CPM)

The **Critical Path Method (CPM)** is a project management tool used to identify the longest sequence of dependent activities that determine the minimum project completion time. For the *Automated Resume Ranking and Role Matching System*, the CPM chart represents the logical flow of major development stages from data collection to testing and deployment.

The process begins with **collecting datasets** and **defining job role categories**, which form the foundation for system design. Subsequent tasks include specifying **hardware and software requirements**, followed by **system architecture design** to establish the overall framework. Once the structure is defined, steps like **data preprocessing**, **AI model integration**, and **feature extraction** are executed to prepare and process the resume data efficiently.

Critical activities such as **model training**, **integration testing**, and **real-time optimization** ensure that the system performs accurately and reliably under varying loads. The **dashboard design** and **UI/UX development** focus on user interaction and visualization of ranked results. Finally, **safety checks** and **documentation** mark the completion of the project.

Through CPM analysis, dependencies among these tasks are clearly mapped, helping to allocate time and resources effectively while identifying critical paths that could impact overall project duration. This structured approach minimizes delays, ensures timely delivery, and enhances workflow efficiency across all development stages.

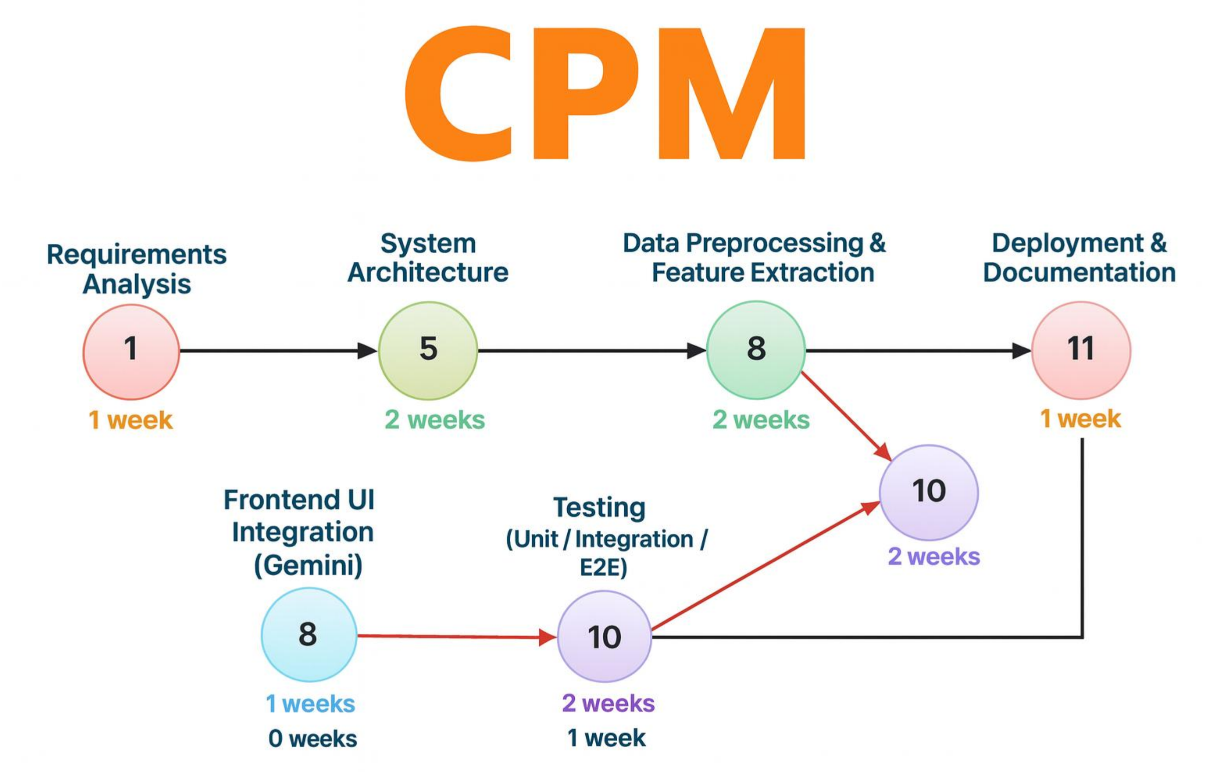
CPM
Requirements Analysis
1
1 week
System Architecture
5
2 weeks
Data Preprocessing & Feature Extraction
8
2 weeks
Deployment & Documentation
11
1 week
10
2 weeks
Frontend UI Integration (Gemini)
8
1 weeks
0 weeks
Testing (Unit / Integration / E2E)
10
2 weeks
1 week

Figure 3 Critical Path

8.13 Program Evaluation and Review Technique (PERT)

The PERT chart illustrates the structured workflow of the *Automated Resume Ranking and Role Matching System* from concept to deployment. The process begins with **requirement gathering and analysis**, followed by designing the **system architecture**. Next, the system moves to **data collection, resume parsing, and AI workflow development**, ensuring accurate extraction of features. The trained model is then **tested for accuracy and performance**, after which it proceeds to **deployment and documentation**. Finally, a **comprehensive evaluation** is performed to assess system reliability, usability, and efficiency before final implementation.

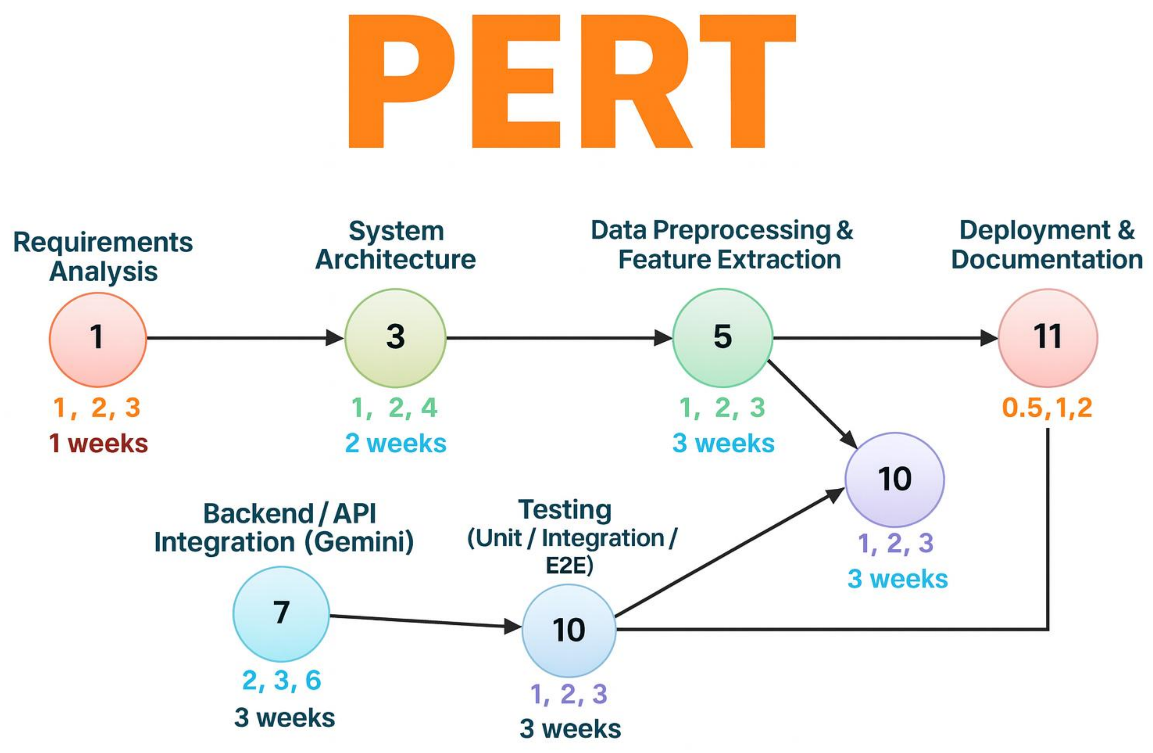

Figure 4 Pert Chart for Design of Automated resume Ranking And Role Matching System

8.2 Staffing

Staffing refers to the process of identifying, allocating, and managing human resources required for successful project execution. It ensures that the right number of people with the right skill sets are available at the right time during various phases of the Software Development Life Cycle (SDLC).

The **objective of staffing** is to balance workload, skills, and timelines efficiently. Proper staffing leads to timely delivery, high-quality outcomes, and cost optimization. In software projects like the *Automated Resume Ranking and Role Matching System*, staffing involves assigning developers, testers, and project coordinators to distinct tasks such as frontend development, backend/API integration, AI/ML modeling, and testing.

8.2.1 Data Flow Diagram (DFD)

The Data Flow Diagram illustrates the overall flow of data within the *Automated Resume Ranking and Role Matching System*. The process begins with users uploading resumes and job descriptions into the system. These inputs are processed through various stages, including **data**

preprocessing and **feature extraction**, where key details like skills, experience, and education are analyzed. The system then performs **resume ranking and scoring** using AI-driven logic to match candidates with suitable job roles. The ranked results are displayed on the **recruiter dashboard**, followed by **report generation** for evaluation and record-keeping. This DFD effectively represents how data moves through the system, ensuring clarity in both functional and operational design.**Result Visualization:** Displays ranked list with "How We Scored" transparency panel.

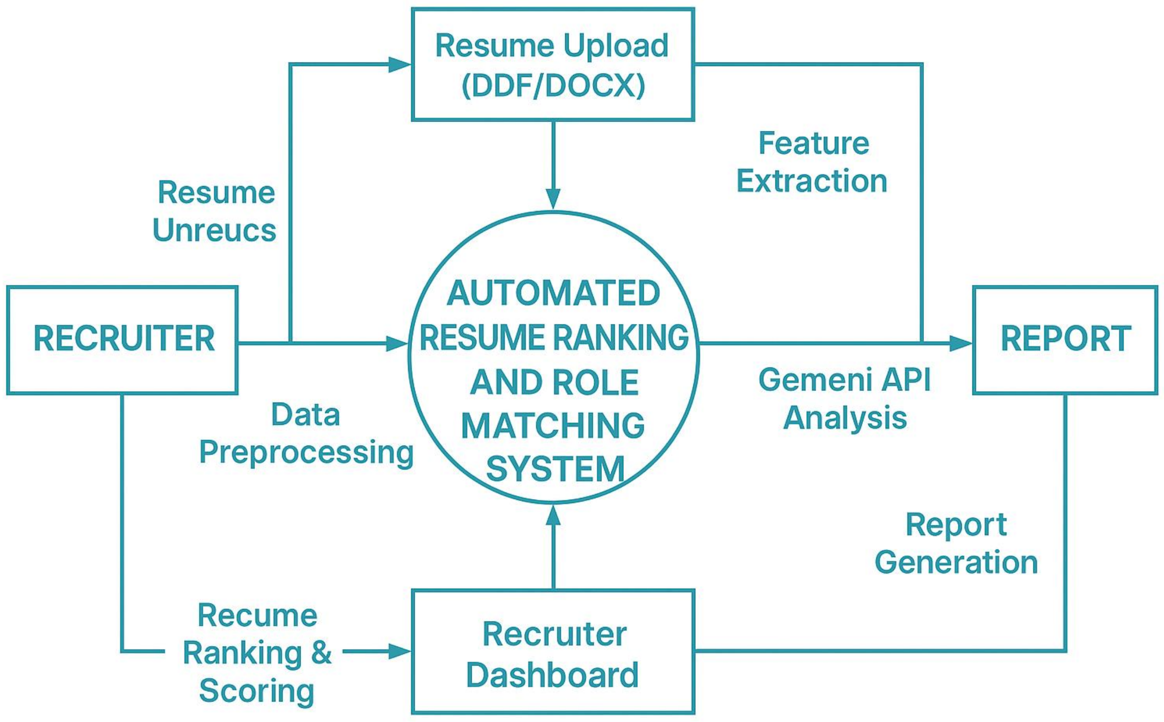

Figure 5 Data Flow Diagram

8.2.2 Function Point Analysis (FPA)

Table 6 Function Point Analysis (FPA)

Component	Count	Complexity	Weight	FP Contribution
External Inputs (Resume Upload, Role Input)	2	Medium	4	8
External Outputs (Ranked List, CSV Export)	2	Medium	5	10
External Inquiries (How We Scored, Blind Mode toggle)	2	Low	3	6

Component	Count	Complexity	Weight	FP Contribution
Internal Logical Files (Temporary Resume DB)	1	Medium	10	10
External Interface Files (Gemini API, Job Role Dataset)	2	High	7	14
Total Unadjusted Function Points (UFP)				**48**

Value Adjustment Factor (VAF): 1.10 (considering AI integration, API interaction, usability)
Adjusted Function Points (AFP):
AFP = UFP × VAF = **48 × 1.10 = 52.8 ≈ 53 FP**

8.2.3 COCOMO Estimation (Basic Model)

Formula:
Effort (Person-Months) = *a × (KLOC)^b*
Time (Months) = *2.5 × (Effort)^0.38*
Where constants depend on project type.

Assumptions:

- Estimated code size ≈ 4.0 KLOC (Python + React + Flask)
- Type: *Organic Project* (small team, familiar domain)
- Constants: *a = 2.4*, *b = 1.05*

Effort = 2.4 × (4)^1.05 = 10.1 Person-Months
Time = 2.5 × (10.1)^0.38 ≈ 6.2 Months
Average Staffing = Effort / Time = 10.1 / 6.2 ≈ 1.6 ≈ 2 persons

Summary

Metric	Estimated Value
Adjusted Function Points	53 FP
Effort (COCOMO)	10.1 Person-Months
Development Time	≈ 6 Months
Team Size	2 Developers + 1 QA (avg)

Appendix A: Glossary

This glossary defines key terms, abbreviations, and acronyms used throughout the document to ensure clarity and consistency.

Term / Acronym	Definition
AI	Artificial Intelligence – The simulation of human intelligence by machines to perform tasks such as decision-making and problem-solving.
NLP	Natural Language Processing – A subfield of AI focused on understanding and processing human language text and speech.
ATS	Applicant Tracking System – Software used by recruiters to collect, sort, and rank job applications automatically.
UI	User Interface – The visual and interactive component through which the user interacts with the application.
UX	User Experience – The overall satisfaction and usability experienced by users when interacting with the application.
API	Application Programming Interface – A software intermediary that allows two applications to communicate (in this case, Gemini API for AI analysis).
Gemini API	Google's AI-powered API used in this system for resume parsing and semantic analysis.
Resume Parsing	The automated process of extracting structured information (skills, experience, education) from resumes.
Semantic Matching	The process of evaluating resumes by meaning and context, rather than simple keyword matching.
CSV	Comma-Separated Values – A common file format for exporting tabular data like ranking metrics.
Frontend	The client-facing layer of the system, responsible for user interaction and visualization (React + TypeScript).
Backend	The server or logic layer responsible for data processing, AI calls, and managing computations.
Blind Mode	A system feature that hides personal identifiers (names, emails, schools) to reduce human bias during candidate evaluation.

Term / Acronym	Definition
Critical Path Method (CPM)	A project management technique used to identify essential tasks that directly affect project completion time.
PERT (Program Evaluation and Review Technique)	A project management tool for analyzing and representing the tasks involved in completing a project, based on time estimation.
WBS (Work Breakdown Structure)	A hierarchical decomposition of the project into smaller, manageable tasks and deliverables.
FR	Functional Requirement – Specifies what the system should do (features, inputs, and outputs).
TP	Test Procedure – The detailed steps or methods used to verify each functional requirement.
NFR	Non-Functional Requirement – Specifies how the system should behave (performance, usability, security, etc.).
HTTPS	Hypertext Transfer Protocol Secure – A secure version of HTTP that uses SSL/TLS encryption for data protection.
WCAG	Web Content Accessibility Guidelines – International standards for web accessibility.
TBD	To Be Determined – Items or details pending final decision or clarification.

Appendix B: Analysis Models

This section includes conceptual and structural analysis models to help visualize system behavior and data flow.

1. Data Flow Diagram (DFD – Level 0)

Description:
The Level 0 DFD illustrates the high-level interaction between system components, data input/output, and the AI processing layer.

Data Flow Summary:

- **Input:** Resume (PDF/DOCX), Job Role Selection
- **Process:** Resume Parsing → Feature Extraction → Semantic Comparison → Ranking
- **Output:** Ranked Results, Match Scores, Visual Dashboard

(Refer to Figure: Data Flow Diagram in Project Report, Chapter 8 for detailed structure.)

2. Use Case Diagram

Actors: Recruiter / Applicant / System (Gemini AI)
Main Use Cases:

- Upload Resume
- Select Job Role
- Analyze Resume
- View Ranked Results
- Export Metrics

(Detailed diagram already included in System Design chapter of report.)

3. Entity-Relationship Diagram (ERD)

Entities and Relationships (Optional Future Enhancement):

- **Entities:** `User`, `Resume`, `JobRole`, `AnalysisResult`, `Metric`
- **Relationships:**
  - A `User` uploads one or more `Resumes`.
  - Each `Resume` is associated with one or more `JobRoles`.
  - Each `AnalysisResult` references both a `Resume` and a `JobRole`.
  - `Metrics` summarize performance and ranking data per session.

(This logical model will be used if the system evolves to include a database layer.)

4. State Transition Diagram (Simplified)

States:

1. **Idle** – Waiting for user input
2. **Uploading** – Resume file being verified
3. **Analyzing** – AI (Gemini API) processing resume
4. **Displaying Results** – Ranked results generated
5. **Exporting Data** – User downloads CSV or report
6. **Complete** – Process ends, system resets

Transitions:
Triggered by user actions (upload, analyze, export) or system events (errors, success).

Appendix C: To Be Determined (TBD) List

The following items are pending confirmation or further research at the time of this SRS submission:

TBD ID	**Item / Description**	**Resolution Plan**
TBD-1	Selection of final **hosting platform** (Vercel, Netlify, or internal university server).	To be finalized after internal testing phase.
TBD-2	Integration method for **batch resume processing**.	Implementation planned in next iteration after MVP validation.
TBD-3	Choice of **database** (PostgreSQL vs Firebase) for persistent storage.	To be determined if system requires permanent data storage.
TBD-4	Final **model bias testing** protocol for ethical compliance.	Awaiting university AI ethics review guidelines.
TBD-5	Design of **multilingual NLP module** for non-English resumes.	Deferred to future scope after English version success.
TBD-6	Detailed **user authentication** feature for enterprise users.	Will be added once system is moved to cloud environment.